

The background of the slide features a close-up, slightly angled view of several wooden Scrabble tiles. The tiles are light brown with black lettering. Visible letters include 'E', 'A', and 'R'. In the bottom right corner, a blue tile with white text reads 'DOUBLE LETTER SCORE'.

ScrabbleAR

Seminario de Python

Julián Feregotti, Andrea Goyechea, Juan Montalivet



Índice general

1	Introducción	4
2	Reglas del ScrabbleAR	5
2.1	Inicio del juego	5
2.1.1	Iniciar partida	5
2.1.2	Configurar juego	5
2.2	Desarrollo del juego	6
2.2.1	Turno del jugador	6
2.2.2	Requisitos para formar palabras en el tablero	6
2.2.3	Posponer partida	6
2.3	Finalización del juego	6
3	Temas estudiados	7
3.1	Sobre PySimpleGUI	7
3.2	Sobre Pattern	8
3.3	Sobre la máquina virtual y el módulo OS	9
3.3.1	La máquina virtual	9
3.3.2	Módulo OS	9
3.4	Sobre entornos virtuales	9
3.4.1	Definición	9
3.4.2	Venv, Virtualenv, Conda	9

4	Problemas y soluciones	10
4.1	Problemas y soluciones surgidos durante el desarrollo	10
4.1.1	El método parse	10
4.1.2	El guardado del tablero	10
4.1.3	La IA	11
5	Consideraciones éticas sobre el desarrollo	12
5.1	Ética Informática	12
5.1.1	Privacidad de la información y Seguridad	12
5.1.2	Uso de recursos y de la propiedad intelectual	13
5.1.3	La brecha digital	13
6	Conclusiones y trabajos futuros	14
6.1	Conclusiones	14
6.2	Trabajos a futuro	14
7	Referencias	16
7.1	Referencias	16
8	Anexo 1: guía de usuario	18
8.1	Instalación del juego	18
8.2	Tablero principal	20
8.2.1	Tableros según el nivel	20
8.2.2	Atril de fichas	20
8.2.3	Información del juego	21
8.2.4	Botones	21
9	Anexo 2: guía para el desarrollador	23
9.1	Formato del tablero de juego	23
9.2	La disposición de fichas	23
9.2.1	Diferenciación de jugadores	24
9.3	Manejo de la configuración	24



1. Introducción

En este informe se detalla el proceso de implementación del juego denominado *ScrabbleAR*¹ desarrollado en la materia Seminario de Lenguajes con opción Python.

ScrabbleAR es un juego de estrategia basado en el popular juego Scrabble, en el que se intenta ganar puntos mediante la asociación de letras para la construcción de distintos tipos de palabras (según el nivel de dificultad) sobre un tablero. Está ideado para un único jugador, el cual deberá competir por puntos contra la computadora.

Lo pueden jugar personas desde los 10 años en adelante, y tiene como objetivo trabajar sobre las habilidades de lógica, matemáticas y léxico. Además, está desarrollado para que todos aquellos que puedan acceder a una computadora con una distribución de Windows o Linux como sistema operativo lo jueguen. Requiere tener instalado python, junto a las principales librerías que utilizamos: PySimpleGUI² y Pattern³.

¹Repositorio del juego: <https://github.com/andr-vg/TrabajoFinalPython>

²Repositorio de PySimpleGUI: <https://github.com/PySimpleGUI/PySimpleGUI>

³Repositorio de Pattern: <https://github.com/clips/pattern>



2. Reglas del ScrabbleAR

2.1 Inicio del juego

Al inicio del juego, se visualiza un menú de opciones para:

- jugar, que permite iniciar una partida nueva, o retomar una partida previamente guardada.
- saber cómo se juega: brinda información del juego y qué tipos de configuraciones y niveles dispone
- configurar el juego: nivel, duración de la partida, cantidad y puntaje por letra.
- visualizar el top 10 con mejores puntajes de partidas previas.

2.1.1 Iniciar partida

- Si se decide iniciar una partida nueva, aparecerá el tablero principal del juego y el primer turno será designado de manera aleatoria, en cualquier caso la primer palabra deberá posicionarse en el centro del tablero.
- Si se retoma una partida anterior, se buscará la información guardada desde un archivo de texto externo y se procederá a mostrar el tablero con la información y las fichas de la partida previa.

2.1.2 Configurar juego

Esta sección permite configurar el juego según preferencias del jugador, entre ellas se encuentran:

- niveles:
 - fácil: se pueden formar sustantivos, adjetivos y verbos.
 - medio: sólo se permiten formar adjetivos y verbos.
 - difícil: se designa aleatoriamente un tipo.
- tiempo: de 1 a 60 minutos.
- puntos por letra: se podrán asignar de 1 a 10 puntos inclusive.
- cantidad de fichas por letra: se podrán asignar de 1 a 10 por letra.

2.2 Desarrollo del juego

El juego dispone de 3 tableros distintos según el nivel de dificultad. Los tableros poseen distintos colores en sus casilleros que indican el tipo, y éstos pueden ser:

- casilleros premio: pueden duplicar o triplicar el puntaje por palabra formada.
- casilleros descuento: pueden restar 1, 2 o 3 puntos al puntaje por palabra formada.
- casilleros normales: no suman ni restan puntos.

Una vez en la ventana principal del juego, se visualiza el tablero, junto a los atriles de la computadora y el jugador, las fichas del jugador con sus respectivos puntajes, y la información necesaria de la partida. Tanto el jugador como la máquina dispondrán siempre de 7 fichas. Éstas se irán reponiendo en el atril a medida que los jugadores formen palabras válidas en el tablero.

2.2.1 Turno del jugador

Durante su turno, el jugador deberá formar una palabra válida con sus fichas y ubicarlas en casilleros libres contiguos del tablero. Si se confunde, puede devolver las fichas al atril (botón deshacer) y comenzar a formar de nuevo. Si no puede formar ninguna palabra con sus fichas, tiene la opción de cambiarlas hasta 3 veces por partida, perdiendo su turno en este caso.

Una vez confirmada la palabra, se analiza si pertenece al tipo de palabra requerido en el nivel, en caso de ser válido, se actualizará el tablero y el puntaje del jugador. Caso contrario, las fichas volverán al atril.

El puntaje por palabra válida se obtiene sumando primero los puntos por cada letra que la forma, y luego, según los casilleros elegidos en el tablero, se duplicará o triplicará este puntaje en caso de ser de tipo premio, o se descontarán puntos en caso de ser de tipo descuento.

2.2.2 Requisitos para formar palabras en el tablero

El jugador debe formar una palabra usando 2 o más letras, colocándolas horizontalmente (las letras ubicadas de izquierda a derecha) o verticalmente (en orden descendente) sobre el tablero. En caso de serle asignado el primer turno, la palabra deberá estar posicionada en el centro del tablero, es decir, una de las fichas deberá ocupar el casillero central. Las palabras no deben cruzarse, pero sí pueden quedar “pegadas” tanto horizontal como verticalmente. Se puede elegir cualquier zona no ocupada del tablero para ubicar la palabra.

2.2.3 Posponer partida

El jugador puede posponer la partida actual. En ese caso se guardará la información del tablero y del juego hasta el momento actual, permitiendo continuar desde este punto la partida en otro momento.

2.3 Finalización del juego

El ganador será el jugador con mayor puntaje al momento de concluir la partida. El juego finaliza cuando:

- se acabó el tiempo
- se acabaron las fichas
- el jugador decidió terminar la partida.



3. Temas estudiados

3.1 Sobre PySimpleGUI

PySimpleGUI es un framework que brinda facilidad de desarrollo, mantenimiento y simpleza a la hora de diseñar una interfaz de usuario.

Su simpleza se basa en la manera en que se construyen los elementos de la interfaz: mediante el uso de listas en Python. El diseño de la ventana es una lista, y a medida que se necesiten nuevas filas, se añaden más listas escribiendo una sola línea de código en el programa. Es decir, el proceso de escribir se vuelve fácil y rápido. La interacción con botones y clics se logra a partir de un bucle `while`. Por último, se pueden personalizar las ventanas de múltiples formas a partir de los parámetros que PySimpleGUI brinda, los cuales son intuitivos y fáciles de comprender. Así como el equipo de PySimpleGUI menciona:

“Simple... keep it simple for the default case. This is part of the PySimpleGUI mission.” [5],

es posible que con dos parámetros ya sea posible personalizar, por ejemplo, un `inputText`. Sin embargo, este también se puede customizar con tantos parámetros como se prefiera.

PySimpleGUI fue utilizado en el trabajo con el propósito de adaptar una interfaz gráfica más agradable al juego que utilizando sólo un terminal, sin requerir de conocimientos avanzados en frameworks de mayor nivel de dificultad, y manteniendo la filosofía del uso de software libre.

La librería soporta actualmente 4 frameworks que se desarrollan y mantienen activamente: Tkinter, Qt con Pyside2-Alpha, WxPython y Remim (soporte para navegadores web, actualmente en desarrollo). Tkinter, que corresponde al framework base de Python, se encuentra soportado en su totalidad. La ventaja de PySimpleGUI frente a estos frameworks es que se pueden definir los mismos widgets, pero de una forma más amigable, con una interfaz simple y eficiente.

Los aspectos de PySimpleGUI que resultaron importantes al momento de diseñar la ventana del juego *ScrabbleAR* fueron:

- el diseño de las ventanas se representa como listas de elementos.

- los valores de retorno son un “evento” tal como pulsar un botón y los valores de entrada corresponden a una lista o diccionario.
- los elementos son clases y los usuarios interactúan con ellos usando métodos de clases, pero no es necesario escribir clases propias para la interfaz.

El equipo de PySimpleGUI aclara que:

“[...] The “Simple” of PySimpleGUI describes how easy it is to use, not the nature of the problem space it solves.[...]”[5]

Esto quiere decir que PySimpleGUI cubre un gran espacio de *problemas de GUI* específicamente, más allá del propósito general que el programador tenga. Esta característica, junto a la preferencia de los programadores de escribir código compacto, permite que su uso sea independiente del nivel de experiencia, de manera que PySimpleGUI está diseñado tanto para principiantes como para desarrolladores experimentados.

En conclusión, PySimpleGUI fue elegido como framework del juego porque brinda una interfaz agradable al jugador, permite que el proceso de aprender a manejar interfaces gráficas de usuario sea mucho más fácil que si se partiera desde otros frameworks, ya que éste permite abstraerse de conocimientos muy avanzados, con el objetivo de centrarse enteramente en la GUI y no en cómo representarla en largas líneas de código.

3.2 Sobre Pattern

Pattern es un módulo de web mining, procesamiento de lenguaje natural, machine learning y análisis de red para Python.

Este trabajo se centró en la función de procesamiento del lenguaje natural para controlar qué tipos de palabras pueden ser ingresadas en el tablero, ya sea por el usuario o la máquina. El módulo cuenta con diferentes opciones de idioma, entre los que se incluyen español, inglés, francés, italiano, alemán y holandés, permitiendo la singularización y pluralización de palabras (con un 94 % de precisión para singularización y 78 % para pluralización), conjugación de verbos, funciones para adjetivos atributivos y predicativos, análisis de sentimientos, que por desgracia no se encuentra implementado en la versión en español, y la función “parse”, la cual es una de las funciones que permiten el funcionamiento correcto del programa. El método parse de pattern nos devuelve una cadena relacionada con las propiedades sintácticas de una palabra. Por ejemplo:

```
> parse('gato'): retorna una cadena del estilo 'gato/NN/B-NP/0'
```

Utilizando la función `split()` sobre lo anterior, ésta retorna la lista `[gato, NN, B-NP, 0]`, de manera que en el trabajo sólo se tomó el valor de la segunda posición, ya que este indica el tipo de palabra, el cual luego se compara con los permitidos según el nivel del juego.

El valor de la segunda posición puede tomar valores como:

- ‘NN’ = Sustantivo.
- ‘VB’ = Verbo.
- ‘JJ’ = Adjetivo.
- ‘RB’ = Adverbio.
- ‘IN’ = Preposición.

Los tipos utilizados en este trabajo fueron ‘NN’, ‘VB’ y ‘JJ’, junto a sus conjugaciones ., para satisfacer las necesidades del juego

3.3 Sobre la máquina virtual y el módulo OS

3.3.1 La máquina virtual

El juego se desarrolló para ser ejecutado en equipos con sistemas Windows y Linux, por lo que se utilizó una máquina virtual para simular una distribución de Linux (Lubuntu) provista por la cátedra y realizar las pruebas. La máquina utilizada fue VirtualBox de Oracle.

“Oracle VM VirtualBox (conocido generalmente como VirtualBox), es un software de virtualización. Actualmente se encuentra desarrollado por Oracle Corporation como parte de su familia de productos de virtualización. Por medio de esta aplicación, es posible instalar sistemas operativos adicionales, conocidos como “sistemas invitados” dentro de otro sistema operativo “anfitrión” y cada uno con su propio ambiente virtual.”[11]

Su uso resultó ventajoso debido a que no fue necesario instalar otro sistema operativo, con la preocupación de llegar a desconfigurar el sistema operativo base.

3.3.2 Módulo OS

Desde las primeras etapas del desarrollo del *ScrabbleAR* se notaron las similitudes y diferencias que ambos sistemas operativos tienen al correr el juego, y los detalles que se debían tener en cuenta para que el programa pueda ser ejecutado correctamente. Por ejemplo, a fin de conseguir una correcta compatibilidad durante la búsqueda de rutas y en la apertura y cierre de archivos tanto en Linux como en Windows, se utilizaron funciones del módulo OS.

3.4 Sobre entornos virtuales

3.4.1 Definición


Un entorno virtual da solución a los inconvenientes que pueden aparecer cuando se necesita utilizar en una o varias aplicaciones distintas versiones de paquetes, sin la necesidad de poseer varias computadoras ni máquinas virtuales.

Según la Python Software Foundation: “tal vez no sea posible para una instalación de Python cumplir los requerimientos de todas las aplicaciones. Si la aplicación A necesita la versión 1.0 de un módulo particular y la aplicación B necesita la versión 2.0, entonces los requerimientos entran en conflicto e instalar la versión 1.0 o 2.0 dejará una de las aplicaciones sin funcionar.”[3]

Se puede definir un entorno virtual como una herramienta que tiene el objetivo de aislar, en un directorio separado, recursos como librerías y entornos de ejecución del sistema principal o de otros entornos virtuales.

3.4.2 Venv, Virtualenv, Conda

Hoy en día existen varios entornos virtuales disponibles, Python (a partir de su versión 3.3) provee como módulo integrado al entorno virtual “venv” en su biblioteca estándar, y permite instalar con pip el entorno virtual “virtualenv” que tiene una api similar al módulo venv, agrega más funcionalidades y funciona desde Python versión 2.6. Como alternativa a venv y virtualenv se encuentra Conda, que es un sistema de manejo de paquetes, dependencias y entornos de código abierto que permite instalar paquetes sin usar pip y puede utilizarse también con otros lenguajes de programación.



4. Problemas y soluciones

4.1 Problemas y soluciones surgidos durante el desarrollo

4.1.1 El método parse

Durante el desarrollo del juego, se observó que el método `parse` de `Pattern` a veces clasifica como sustantivos a palabras que no existen o no encuentra. Por ejemplo, si escribimos:

```
> parse('asdas')
```

Este devuelve: `'asdas/NN/..'`, correspondiente a un sustantivo ("NN"). La dificultad frente a esta situación se encuentra en que tanto el usuario como el algoritmo de la máquina podrían ingresar cualquier combinación de letras para generar palabras y se tomarían como un sustantivo válido.

La solución que se propuso fue la de agregar un chequeo adicional donde se pregunte si la palabra generada se encuentra tanto en `Spelling` y `Lexicon`, como en `Verbs` en el caso de verbos. Estos archivos son propios del módulo `pattern.text` y contienen palabras junto a un conjunto de 'tags', de manera que, si la palabra no se encuentra en ninguno de estos archivos, ésta se tome como inválida y el jugador o máquina deban formar otra. Para el chequeo se utilizó una función que es una adaptación a un ejemplo que hizo un ex alumno en sus notas sobre la librería de `Pattern`¹.

4.1.2 El guardado del tablero

Durante la mitad del desarrollo del juego se utilizó el formato de archivo CSV para los tableros, ya que este en un principio parecía más práctico, debido a que su disposición es similar al tablero que se visualiza posteriormente en el juego. La forma en que se llevaba a cabo esta implementación era la siguiente:

Se disponía de un tablero csv con los símbolos "+", "-", "++" y "--" para indicar el tipo de casillero (premio o descuento), cuyas posiciones se correspondían con las del tablero creado con

¹Repositorio del alumno Juan Pablo: https://github.com/pibytes/notas_python/blob/master/patternES.md

PySimpleGUI. Se leía el csv fila por fila en una lista, y cada símbolo se traducía en un casillero de la interfaz.

El problema surgió al momento de hacer el proceso inverso, que consistía en traducir el contenido de esos casilleros a una lista para guardarlos posteriormente en un CSV cuando el jugador quería posponer una partida. Debido a limitaciones propias del módulo, algunas de las claves que correspondían a las posiciones se guardaban con un formato diferente al especificado (se le agregaba un símbolo ‘*’ de más), provocando errores tales como `keyError` al momento de cargar la partida pospuesta, o perdíamos desde posiciones intermedias hasta incluso la mitad del tablero. Al no encontrarle una solución, se optó por cambiar el formato de los tableros a JSON, donde las claves corresponden a las coordenadas de los botones en el tablero, y los valores a las fichas.

A partir de esto, como JSON se asemeja a un diccionario de Python, no hubo más problemas en la carga y guardado del mismo. Sin embargo, se encontró tedioso el hecho de tener que armar los 3 tableros personalizados del juego, ya que no se podía visualizar cómo iba a quedar a medida que los armábamos. Frente a esta situación, un integrante realizó un programa que permite dibujar el tablero como si fuera un “Paint”². con las paletas de colores elegidas según la dificultad, y luego exportarlo de manera inmediata a un formato JSON para poder finalmente utilizarlo en el juego principal.

4.1.3 La IA

La inteligencia proporcionada a la máquina consiste en:

- reconocer espacios libres en el tablero donde pueda formar palabras.
- formar palabras con sus fichas y las posiciones libres.

La dificultad implementada inicialmente utilizaba un recorrido secuencial para obtener casilleros dispuestos juntos y libres, formaba un conjunto de palabras posibles y se quedaba con la de mayor longitud que quepa en los casilleros. La elección de los casilleros libres y la forma en que se posicionaba cada letra era aleatoria.

A medida que fuimos desarrollando los distintos niveles de dificultad, encontramos que la inteligencia de la máquina y su jugabilidad dependían exclusivamente de la aleatoriedad, sin mucha capacidad de razonamiento lógico. Frente a esto, el contrincante siempre ganaba.

Ante este problema decidimos subdividir la IA por niveles:

- fácil: se continuó con la aleatoriedad descrita anteriormente.
- medio: se dotó a la IA de cierta lógica que le permita reconocer posiciones contiguas que al menos tengan un casillero con premio.
- difícil: la IA además obtiene un diccionario de posiciones cuya clave contiene la cantidad de casilleros premio y de estas elige la mejor. Luego realiza un análisis lógico que le permite seleccionar posiciones que le permitan (de ser posible) duplicar o triplicar la letra de mayor puntaje que posea su palabra, o quedarse con las posiciones que dupliquen o tripliquen palabra (si es que las tiene), según el puntaje obtenido en ambas.

También proporcionamos como complemento de los métodos en niveles medio y difícil el uso del método fácil para el caso en que la máquina no encuentre casilleros con premio.

²Repositorio del alumno J. Feregotti: <https://github.com/julianfere/Herramienta-tablero-ScrabbleAR>



5. Consideraciones éticas sobre el desarrollo

5.1 Ética Informática

Según la cátedra de Ética, Legislación y Profesión de la Facultad de Informática de la Universidad Complutense de Madrid:

“En términos generales, la Ética Informática es la disciplina que analiza problemas éticos que son creados por la tecnología de los ordenadores, también los que son transformados o agravados por la misma.” [4]

Esto quiere decir que, cuando hablamos de la ética informática, nos referimos a un campo que engloba no sólo el proceso de creación y desarrollo del software o hardware, sino también su posterior transformación y el impacto que éste genera en la sociedad.

Este estudio pone al descubierto los problemas morales sobre la privacidad de la información de los usuarios, problemas que surjan durante el ciclo vital (creación, grabación, distribución), la propiedad y el copyright, la brecha digital, entre otros.

5.1.1 Privacidad de la información y Seguridad

Hoy en día, la informática se volvió parte de nuestras vidas y un problema importante está en la manera en que nuestra información puede verse comprometida al ser cedida a las aplicaciones para ser compartidas públicamente. Desde crear una aplicación de escritorio hasta una aplicación web. Si bien está en el propio usuario la decisión de compartir su información, desde un punto de vista ético, esta podría llegar a ser vulnerada por un eventual ataque en la privacidad. Además, a fin de buscar que los anuncios encajen con sus criterios de búsqueda, el usuario termina siendo visto más como un “producto”.

Como desarrollador, también es necesario que al momento de crear un software se trabaje para evitar que luego de su distribución, éste se vea agravado por fines maliciosos de otras personas o afecte a un sector determinado de la población (ej: cyberbullying).

Por esto, es importante definir y reconocer problemas éticos en la informática para poder actuar lo más rápido posible y evitar vulnerar al usuario.

5.1.2 Uso de recursos y de la propiedad intelectual

Otro objetivo dentro de la ética informática se basa en la utilización de los recursos informáticos de manera adecuada, sin tomar como propios los trabajos realizados por otras personas. Ni tampoco copiar y redistribuir software sin permisos del propietario. Para el desarrollo del juego, se utilizaron recursos que corresponden a software libre. Esto significa que su código fuente puede ser usado, estudiado, distribuido y mejorado sin infringir la licencia. *ScrabbleAR* también lo es. Por otro lado, las imágenes que se utilizaron fueron creadas por los propios integrantes de este informe.

5.1.3 La brecha digital

Un problema presente principalmente en los países en desarrollo y subdesarrollados corresponde a la brecha digital generada por el alto costo de las Tecnologías de la Información y la Comunicación (TIC), de manera que su obtención, conocimiento y uso se realiza de manera desigual en la población. También existen factores geográficos, geopolíticos, culturales, de género, entre otros.

El acceso al conocimiento debería ser libre, y reducir la división digital debe ser un objetivo importante dentro de la ética informática, porque al final, por más que todo el software y la información sean libres, de nada sirve que una persona pueda acceder a estas herramientas si en su casa no dispone de algo tan básico como una computadora.



6. Conclusiones y trabajos futuros

6.1 Conclusiones

En este informe detallamos los conocimientos aprendidos que fueron utilizados para el proyecto, la forma en que aplicamos las pautas de cómo debería funcionar el juego, y los problemas que fuimos encontrando y resolviendo.

Durante el desarrollo de este trabajo, fuimos experimentando las diferentes situaciones que conllevan preparar, realizar y dar un cierre a un proyecto grupal. El hecho de que hayamos encarado el trabajo con compañeros de curso, en el marco de la materia de Seminario de Lenguajes y utilizando las herramientas explicadas (Python, Github, entre otras), hizo que la experiencia obtenida por los integrantes del grupo desarrollador haya sido valiosa como aprendizaje personal, y desde nuestro punto de vista digna de compartir.

Haciendo una vista general de lo que fue este trabajo en grupo y a modo de reflexión, llegamos a darnos cuenta que a medida que se iba programando el juego, fuimos aprendiendo características nuevas sobre las herramientas que utilizamos, facilitándonos tareas que antes nos parecían complicadas, y dejándonos un conocimiento muy útil para futuros proyectos y aplicaciones.

6.2 Trabajos a futuro

Los tópicos a realizar en el futuro son los siguientes:

- utilización de librerías soporte y algoritmos que permitan filtrar mejor palabras que no correspondan a los tipos especificados en los distintos niveles del juego.
- permitir que las palabras con tilde se tomen como válidas.
- implementación de algoritmos para el turno de la computadora cuya selección de posiciones sea aun más acertada y realizando un mapeo que encuentre la mejor posición con el menor error posible. Es necesario desarrollar una estadística más fina a fin de poder establecer mejor una

relación palabra-puntaje-posiciones libres. El jugador humano, al estar dotado de cierta lógica implícita encuentra esta relación sin tantos problemas. Por el contrario, los algoritmos usados en este trabajo se enfocaron en obtener palabras más largas primero. Como consecuencia, en algunas circunstancias nos encontramos con la dificultad de no poder tomar una palabra quizás más corta pero que suma más puntos, afectando de esta forma a la inteligencia de la máquina.

- modificación en la forma de ubicar las fichas, permitir formar palabras con fichas ya puestas en el tablero como sucede en el juego original Scrabble.
- brindar botones que permitan al jugador acceder en determinados momentos del juego a una “ayuda” o “sugerencia” para formar palabras, según el nivel y con una disponibilidad de accesos finita, como sucede con el botón cambiar fichas.
- incursionar en las tecnologías de socket y multi-threading de python para agregar la opción de multijugador en una red local.



7. Referencias

7.1 Referencias

- [1] Python Software Foundation. *OS - Miscellaneous operating system interfaces*. 2001-2020. URL: <https://docs.python.org/3/library/os.html>.
- [2] Python Software Foundation. *os.path - Common pathname manipulations*. 2001-2020. URL: <https://docs.python.org/3/library/os.path.html>.
- [3] Python Software Foundation. *Virtual Environments and Packages*. 2020. URL: [URL: %20https://docs.python.org/es/3/tutorial/venv.html](https://docs.python.org/es/3/tutorial/venv.html) (véase página 9).
- [4] Facultad de Informática de la Universidad Complutense de Madrid. *La Ética Informática*. 2016-2019. URL: http://wikis.fdi.ucm.es/ELP/La_%C3%89tica_Inform%C3%A1tica (véase página 12).
- [5] The PySimpleGUI Organization. *PySimpleGUI User's Manual*. URL: <https://pysimplegui.readthedocs.io/en/latest/> (véanse páginas 7, 8).
- [6] Walter Daelmans Tom De Smedt. *Pattern: Natural Language Processing*. URL: <https://github.com/clips/pattern>.
- [7] Penn Treebank. *Penn Treebank II Tags*. URL: <https://web.archive.org/web/20130517134339/http://bulba.sdsu.edu/jeanette/thesis/PennTags.html>.
- [8] La enciclopedia libre Wikipedia. *Brecha Digital*. 2020. URL: https://es.wikipedia.org/wiki/Brecha_digital.
- [9] La enciclopedia libre Wikipedia. *Infoética*. 2020. URL: <https://es.wikipedia.org/wiki/Info%C3%A9tica>.
- [10] La enciclopedia libre Wikipedia. *Software Libre*. 2020. URL: https://es.wikipedia.org/wiki/Software_libre.

-
- [11] VirtualBox Wikipedia. 2020. URL: <https://es.wikipedia.org/w/index.php?title=VirtualBox&oldid=129675923> (véase página 9).

8. Anexo 1: guía de usuario

En esta sección se visualiza al usuario las distintas ventanas y botones correspondientes al juego, para un mejor entendimiento del programa.

8.1 Instalación del juego

Dentro de la carpeta *ScrabbleAR*, deberá ejecutar:

```
> python3 -m pip install -r requirements.txt
```

para poder instalar las dependencias del juego. Para ejecutar el juego, dentro de la carpeta *bin* deberá correr:

```
> python3 ScrabbleAR.py
```

y se mostrará la ventana de inicio del juego.

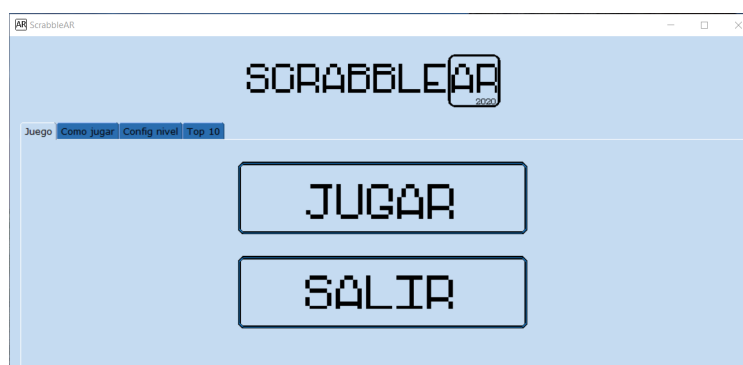


Figura 8.1: Menú principal.

Dentro de esta ventana, se puede acceder a las reglas para entender cómo se juega, observar y modificar la configuración actual del juego y visualizar el top 10 de las mejores partidas según el nivel de dificultad.

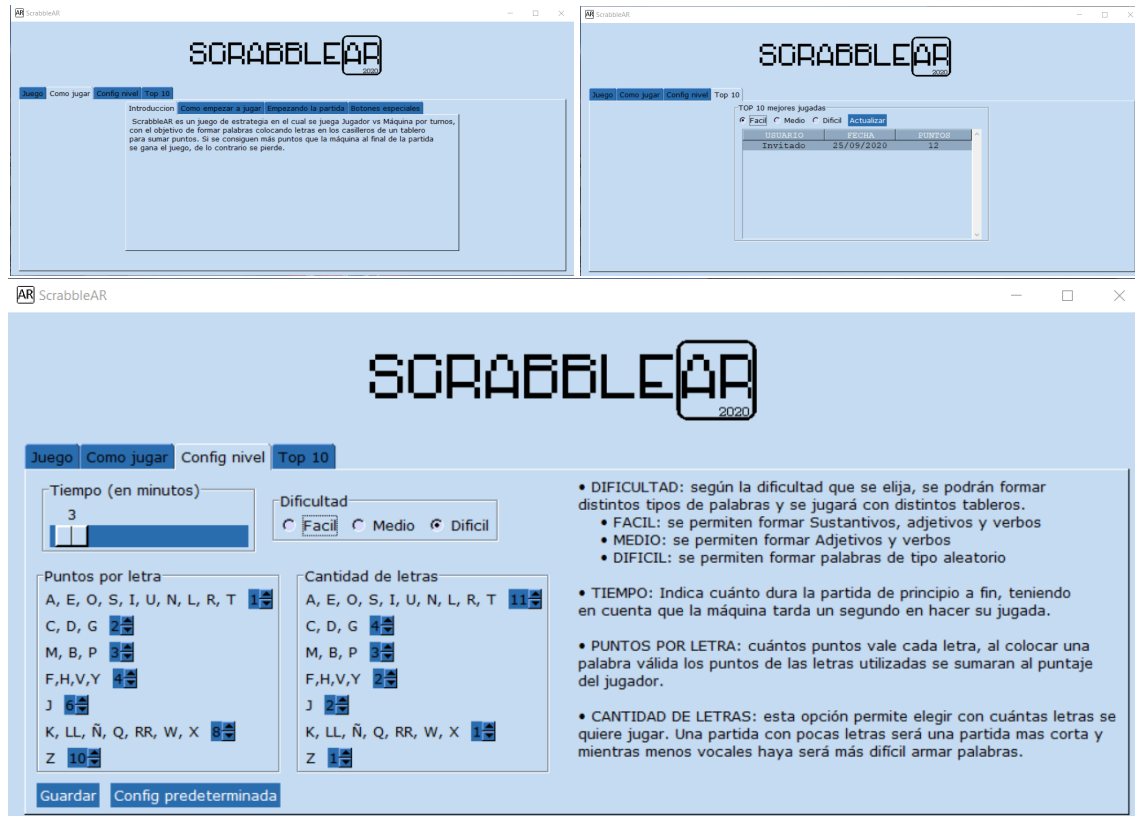


Figura 8.2: Ayuda al usuario, top 10 y configurar el juego.

Es posible modificar el tiempo, la dificultad del juego, el puntaje a obtener por grupos de letras, así como también la cantidad de letras que habrá en la bolsa.

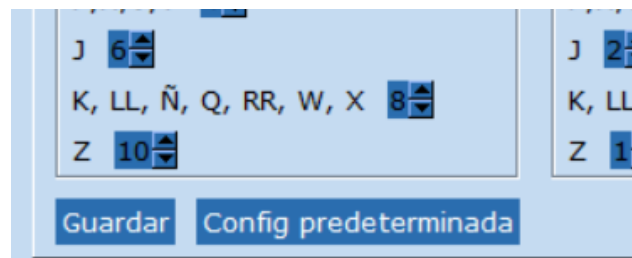


Figura 8.3: Visualización de los botones en configuración del nivel.

Como se puede observar en la figura 8.3, el botón “guardar” configura la próxima partida con las opciones elegidas por el usuario. Por otro lado, si se quiere retornar a la configuración predeterminada del juego, es necesario clickear el botón “Config. predeterminada”.

8.2 Tablero principal

Una vez elegida la configuración deseada, de acuerdo a la dificultad seleccionada se visualiza uno de los 3 tableros disponibles.

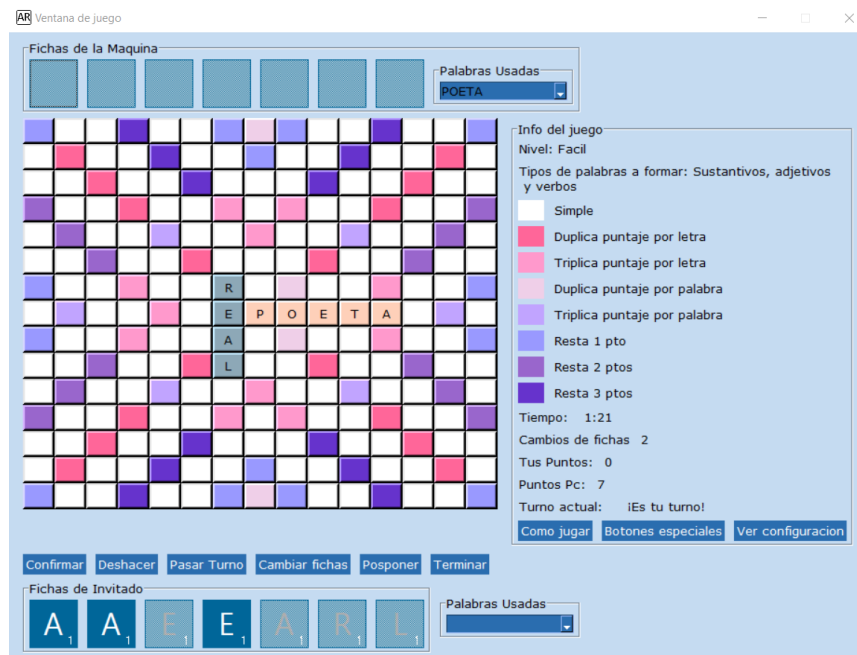


Figura 8.4: Ventana de la partida actual.

8.2.1 Tableros según el nivel

La información que provee el juego junto a la customización y palabras a formar en el tablero variará de acuerdo a las configuraciones elegidas en la ventana inicial.

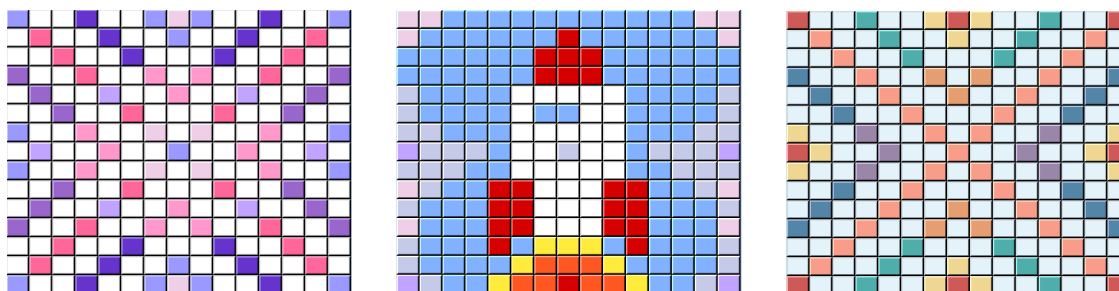


Figura 8.5: Personalización de los tableros en los niveles fácil, medio y difícil.

8.2.2 Atril de fichas

Las fichas asignadas a los jugadores serán siempre 7, y dentro de cada una se puede ver la letra y el puntaje que esta suma indicado con un subíndice en la esquina inferior derecha de la ficha. Junto

al atril, se encuentra una lista desplegable con las palabras válidas que el jugador va formando a lo largo de la partida.



Figura 8.6: Atril del usuario.

La máquina también posee un atril y lista de palabras formadas, pero durante el desarrollo de la partida no será posible visualizar sus fichas.

8.2.3 Información del juego

La información que provee el tablero durante el juego corresponde a:

- dificultad del juego y tipos de palabras que se pueden formar.
- indicaciones de los distintos tipos de casilleros (simple, premio y descuento) según su color.
- tiempo restante de la partida actual en min:seg.
- número de cambios de fichas disponibles.
- puntaje actual de los usuarios.
- turno actual.

El usuario también puede acceder a una ayuda durante la partida a través de los botones que se encuentran debajo de la sección de información del juego.

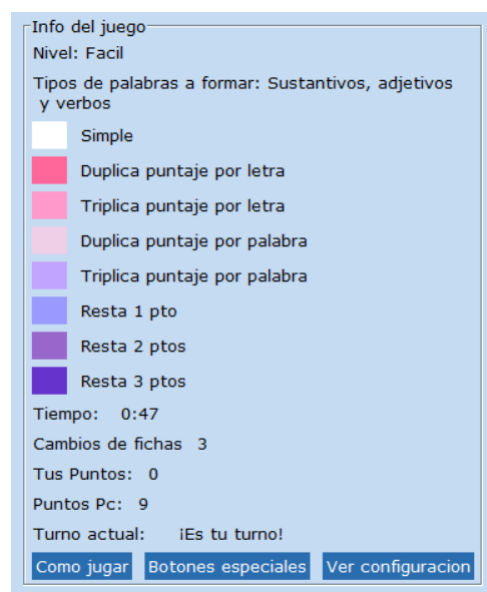


Figura 8.7: Información provista durante la partida.

8.2.4 Botones

En el transcurso de su turno, el usuario puede clicar botones que le permiten acceder a distintas opciones provistas en el juego. Entre ellos se encuentran:

- Botón “Confirmar”: sólo se encuentra disponible si el usuario ya ingresó al menos 2 letras en el tablero. Una vez seleccionado, el juego analiza si las fichas ingresadas forman una palabra válida. De ser así, se calcula y actualiza el puntaje, se reemplazan las fichas usadas y se pasa el turno a la máquina. En el caso contrario, las fichas retornan al atril y el usuario debe volver a intentarlo.
- Botón “Deshacer”: se habilita después de que el usuario selecciona la primer ficha. Este le permite devolver todas las fichas que eligió al atril y volver a comenzar.
- Botón “Pasar Turno”: si lo desea, puede elegir perder el turno actual y dárselo a la máquina.
- Botón “Cambiar fichas”: sólo se puede utilizar 3 veces durante el juego, y permite al usuario elegir las fichas que quiere intercambiar con la bolsa (desde 1 a todas las del atril). El usuario perderá el turno.

- Botón “Posponer”: en cualquier momento se puede guardar¹ la partida en su estado actual y jugar en otro momento. Una vez seleccionado el juego se redirige al menú inicial. Para volver a jugar a dicha partida, desde el menú principal se le preguntará si desea continuar la partida o iniciar una nueva.
- Botón “Terminar”: una vez presionado, el juego preguntará si se desea dar por finalizada la partida o seguir jugando. Si se decide dar por finalizado el juego actual, se recalcularán los puntajes, se hará visible el atril de la máquina y se informarán los resultados finales.

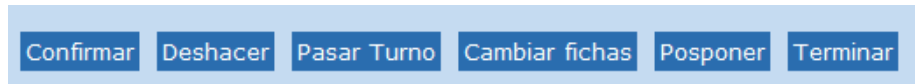


Figura 8.8: Botones disponibles.

¹Aclaración: Se puede guardar una sola partida. Si ya hay otra, esta será sobreescrita.



9. Anexo 2: guía para el desarrollador

9.1 Formato del tablero de juego

Como se especificó en el apartado de “Problemas y soluciones surgidas durante el desarrollo”, el formato de los tableros es JSON y tienen la siguiente estructura: “Coordenada:Tipo de ficha”.

Por ejemplo:

```
{  
  (0,0): "+",  
  (0,1): " ",  
  (0,2): "-",  
  (0,3): "--",  
  (0,4): "---",  
  (0,n): "++"  
}
```

Listing 1: JSON del tablero

De esta forma, se pueden realizar tableros tanto rectangulares como cuadrados.

9.2 La disposición de fichas

```
{  
  ' ' : Ficha simple,  
  '+' : Duplica puntaje por letra,  
  '++': Triplica puntaje por letra,  
  '+++': Duplica el puntaje de la palabra,
```

```

    '-' : Resta un punto,
    '--' : Resta dos puntos,
    '---' : Resta tres puntos,
}

```

9.2.1 Diferenciación de jugadores

Para diferenciar las letras entre el jugador y la máquina se utilizó la siguiente simbología:

- Letra '*': letra de la maquina.
- Letra '/': letra del jugador.

9.3 Manejo de la configuracion

A la hora de configurar el juego se establecieron distintas opciones preestablecidas (localizadas en el archivo *configPred.json*), tales como: cantidad de fichas y puntos de las mismas. En este programa la disposición está diseñada por grupos y en el JSON de configuraciones predeterminadas encontrará lo siguiente:

```

{
  "tiempo": "3:00",
  "dificultad": "facil",
  "grupo_1": 1,
  "grupo_2": 2,
  "grupo_3": 3,
  "grupo_4": 4,
  "grupo_5": 6,
  "grupo_6": 8,
  "grupo_7": 10,
  "grupo_1_cant": 11,
  "grupo_2_cant": 4,
  "grupo_3_cant": 3,
  "grupo_4_cant": 2,
  "grupo_5_cant": 2,
  "grupo_6_cant": 1,
  "grupo_7_cant": 1
}

```

"grupo_i" (i=1..7) corresponde al puntaje para las letras de cada grupo, mientras que "grupo_i_cant" (i=1..7), contiene la cantidad de repeticiones que tendrán esos grupos de letras. Los grupos se definen de la siguiente manera:

- Grupo 1: "A, E, O, S, I, U, N, L, R, T"
- Grupo 2: "C, D, G"
- Grupo 3: "M, B, P"
- Grupo 4: "F, H, V, Y"
- Grupo 5: "J"
- Grupo 6: "K, LL, Ñ, Q, RR, W, X"
- Grupo 7: "Z"